Optimal Cooperation and Submodularity for Computing Potts' Partition Functions with a Large Number of States

J-Ch. Anglès d'Auriac[*]

F. Iglói[†]

M.Preissmann and A.Sebő[‡]

February 1, 2008

Abstract

The partition function of the q-state Potts model with random ferromagnetic couplings in the large-q limit is generally dominated by the contribution of a single diagram of the high temperature expansion. Computing this dominant diagram amounts to minimizing a particular submodular function. We provide a combinatorial optimization algorithm, the optimal cooperation algorithm, which works in polynomial time for any lattice. Practical implementation and the speed of the method is also discussed.

Introduction

Spin models with discrete symmetry are often used to describe order-disorder transitions, such as in magnetic systems or in absorbed monolayers, etc, for a review see in[1]. In two-dimensional regular lattices and with homogeneous

[*]CNRS-CRTBT B. P. 166, F-38042 Grenoble, France

[†]Research Institute for Solid State Physics and Optics H-1525 Budapest, P.O.Box 49, Hungary and Institute for Theoretical Physics, Szeged University, H-6720 Szeged, Hungary

[‡]Laboratoire LEIBNIZ-IMAG 46, Avenue Félix Viallet, 38000 Grenoble Cedex, France

couplings there are some models (Ising model, critical Potts model, eight-vertex model, etc.) with an asymptotically exact solution for the partition function in the thermodynamic limit[2]. In the presence of quenched, that is time independent, disorder this type of exact solutions are scarce which has greatly hampered our understanding about collective phenomena in disordered systems, for example about the properties of spin glasses.

For disordered systems, such as for spin models with random couplings and/or fields one usually performs an exact computation on finite lattices and the results are then extrapolated. An important limitation of this finite lattice method is the available sizes, which can be treated by different computational methods. For most of the discrete spin models a computation of the ground state represents a hard optimization problem when there is frustration and it is even much more complex to compute the partition function.

In this respect the Ising model, which is equivalent to the Potts model with $q = 2$ states, is an exceptional problem since the ground (minimum energy) states can be found in polynomial time using the solution of max cut problems in planar graphs that involves matching theory [3] [4] [5]. The full partition function of the Ising problem can also be computed for lattices of bounded genus [6]. Another example of application of optimization theory to lattice statistical mechanics is the Random Field Ising Model where the ground state can be found in polynomial time in any dimension [7] [8]. A discussion about optimal cuts in statistical mechanics can be found in [9] and a general review about the application of combinatorial optimization methods in statistical physics can be found in[10].

For the $q = 3$-state Potts model, when the number of states is three, the computation of the minimum energy of the system involves (NP-)hard combinatorial problems such as minimizing the 3-cut in a graph, and no efficient procedure is known for computing the partition function.

If the number of states can be arbitrary large (and given as part of the input) the partition function has an exponential number of terms, and the problem may look hopeless. In this paper we show that this is not the case: when the number of states tends to infinity, the partition function can be determined up to an integer multiplicative factor, and some average properties of the system can be inferred. This evaluation goes trough the minimization of a submodular function. Note that in [11] a simulated annealing method and an approximate combinatorial method have been used. The solution we

2

give here is both exact and much more efficient.

Even though there are ready-made algorithms for general submodular function minimization [12] [13] [14] [15], the resources they require are far beyond the possibilities of the present application. While the existing first implementations of general submodular function minimization may require too much time already to solve the optimization problem on a lattice containing a dozen of sites (vertices of the input graph corresponding to the variables of the general problem), we want and will solve problems involving more than two hundred thousand 'sites'. For this we use methods inspired by [16] [17] [18] [19] that fit the particularity of the application and develop an algorithm which is tailored to graphic submodular functions. Besides the computational advantages of this approach, we are also aiming at conceptual simplicity and elegance. Readers who seek for more mathematical background and connections can consulte [20].

In Section 1 we describe the problem of physics and show how it reduces to a function maximization. In Section 2 we formulate the mathematical problem, the preliminaries and the broader context of our algorithm; in Section 3 and 4 we show how to construct an optimal solution and in Section 5 we state the algorithm. Finally, in Section 6 we analyze the practical implementation of the method and display a few computational results. The paper is self-contained, the algorithms and their correctness will be fully proved.

1   The problem in terms of physics.

For a review of the huge amount of work devoted to the Potts model see [21]. In this section we recall the Potts model and show why its partition function (up to an integer multiplicative factor) is determined, if the number of 'states' tends to infinity by the minimum value of a submodular function.

A *lattice* of spins is given at each *site* of which is a variable. We denote the number of sites by n. Each variable $\sigma_1, \ldots, \sigma_n$ can take values in $\{0, 1 \ldots, q-1\}$ for a given integer q. Pairs of neighboring sites on the lattice are called *bonds*, the number of bonds by m.

Each configuration $\sigma = (\sigma_1, \ldots, \sigma_n)$ has an energy

$$E(\sigma) = \sum_{ij} K_{ij} \delta_{\sigma_i \sigma_j}, \tag{1}$$

3

where $\sigma_i \in \{0, \ldots, q-1\}$ the sum runs over all bonds ij, $K_{ij} \in \mathbb{R}_+$ is a given non-negative weight of bond ij, and δ_{ab} is the Kronecker symbol (1 if $a = b$ and 0 otherwise). For mathematicians: a lattice is a graph, sites are vertices, bonds are edges. The aim is to compute or approximate the partition function

$$Z(K_{ij} : ij \text{ is a bond}) = \sum_\sigma \exp(E(\sigma)) = \sum_\sigma \prod_{ij} e^{K_{ij}\delta_{\sigma_i\sigma_j}}, \qquad (2)$$

where the summation runs over all assignments of values $\sigma = (\sigma_1, \ldots, \sigma_n)$, and the product over all bonds ij (the inverse temperature has been included in the couplings K_{ij}).

We follow [11]. Note that $e^{K\delta} = 1 + (e^K - 1)\delta$ for $\delta \in \{0, 1\}$; introduce $\nu_{ij} = e^{K_{ij}} - 1$ and expand the product of sums:

$$\prod_{ij} e^{K_{ij}\delta_{\sigma_i\sigma_j}} = \prod_{ij}(1 + (e^{K_{ij}} - 1)\delta_{\sigma_i\sigma_j}) = \sum_F \prod_{ij \in F} \nu_{ij}\delta_{\sigma_i\sigma_j},$$

where the summation runs over all subsets of bonds F. Substituting this to (2) we get:

$$Z(K_{ij}) = \sum_\sigma \sum_F \prod_{ij \in F} \nu_{ij}\delta_{\sigma_i\sigma_j}$$

$$= \sum_F \sum_\sigma \prod_{ij \in F} \nu_{ij}\delta_{\sigma_i\sigma_j} = \sum_F q^{c(F)} \prod_{ij \in F} \nu_{ij}, \qquad (3)$$

where the sum runs over all subsets F of bonds, and where $c(F)$ is the number of connected components of F on the set of all sites, counting also the isolated sites among the components. (By convention if F is empty then $\prod_{ij \in F} \nu_{ij} = 1$.) We got the last equality by counting the number of different $\sigma = (\sigma_1, \ldots, \sigma_n)$ for which the product is nonzero; since it is nonzero if and only if $\sigma(i) = \sigma(j)$ for every $ij \in F$, that is, if and only if σ is constant on every connected component of F, all possible σ can be enumerated by choosing an element of Z_q for every connected component of F independently. Therefore we have exactly $q^{c(F)}$ such σ.

Note that this sum has 2^m terms, and that in (3) q does not need to have an integer value. (This provides a way of defining Potts' model for non-integer values [22].) Clearly, $K_{ij} \geq 0$ is equivalent to $\nu_{ij} \geq 0$, and we can introduce a new set of variables α_{ij} with

$$\nu_{ij} = q^{\alpha_{ij}},$$

4

and the partition function becomes

$$Z = \sum_F q^{c(F)+\sum_{ij \in F} \alpha_{ij}}. \tag{4}$$

Finally one introduces the function

$$f(F) = c(F) + \sum_{ij \in F} \alpha_{ij} \tag{5}$$

so that

$$Z = \sum_F q^{f(F)}. \tag{6}$$

As pointed out in ref [11], while q tends to infinity the sum in (6) is asymptotically equal to $Z = Nq^{f^\star}$ where $f^\star = max_{F \subseteq \{1,...,n\}} f(F)$ and N is the number of optimal sets. Note that if the weights w_{ij} are arbitrary reals, the degeneracy N is likely to be 1.

We will analyze the combinatorial optimization problem of finding this maximum. The problem of finding the degeneracy of the optimal set will not be addressed. In the analysis of the Potts model magnetic properties are related to geometrical properties of the optimal sets A, and specifically to their fractal dimension (if any). We assume that, on average and for large enough lattice, this fractal dimension is the same for all sets, and therefore finding a single optimal set is sufficient for our purpose. We will present methods which apply to any set of weights $\{\alpha_{ij}\}$ in an arbitrary lattice (graph) and provide a simply stated and efficient algorithm.

2   The problem in terms of graphs.

For basic graph theoretic notions, terminology and notation we refer to [23]. Given a graph $G = (V, E)$ on n vertices with weight function $w : E(G) \longrightarrow \mathbb{R}$, we want to solve the following problem:

$$\text{Maximize } \{ f_{G,w}(A) = c_G(A) + \sum_{e \in A} w(e) : A \subseteq E(G) \} \qquad (POTTS),$$

where $c_G(A)$ is the number of connected components of the graph $G(A) = (V, A)$. Note a slight change of notation with Eq. 5 where an edge is denoted by ij the two corresponding adjacent vertices. We can suppose $0 < w(e) < 1$

for all $e \in E(G)$, because if $w(e) = 0$, then deleting the edge, if $w(e) \geq 1$, then contracting it (identifying its two endpoints) we get an equivalent problem.

When no confusion is possible we will simply use f or f_G for $f_{G,w}$, c for c_G. This function that has to be maximized in order to solve (POTTS) has a crucial and well-known property (see for instance [23], Exercise 6.2). For any two subsets A and B of edges of G :

$$f(A) + f(B) \leq f(A \cup B) + f(A \cap B). \qquad (SUPER)$$

Indeed, it is clear that $\sum_{e \in A} w(e) + \sum_{e \in B} w(e) = \sum_{e \in A \cup B} w(e) + \sum_{e \in A \cap B} w(e)$, so we only have to show that $c(A) + c(B) \leq c(A \cup B) + c(A \cap B)$; that is c satisfies the (SUPER) property. For the sake of completeness we include a proof. Proceed by induction on $|A \setminus B| + |B \setminus A|$. If this number is 0 then the statment is obvious. If not, then there exists say $e \in A \setminus B$. By induction $c(A \setminus e) + c(B) \leq c(A \cup B \setminus e) + c(A \cap B)$. Deleting an edge increases the number of connected components by at most 1 and so we are done unless the equality holds and $c(A \setminus e) = c(A)$ and $c(A \cup B \setminus e) = c(A \cup B) + 1$. But this is impossible since $c(A \setminus e) = c(A)$ means that the two endpoints of e are in the same connected component of $G(A \setminus e)$, and then also in the same component of $G(A \cup B \setminus e)$.

A function g defined on the set of subsets of a set S, with values in $\mathbb{R}$ is said to be *submodular* if $g(A) + g(B) \geq g(A \cup B) + g(A \cap B)$ for all subsets A and B of S. The (SUPER) property shows that the function $-f$, whose minimization solves (POTTS), is submodular.

By now several polynomial algorithms [12] [13] [14] [15] are known to minimize a submodular function, therefore we can already claim that there exists a polynomial algorithm solving our problem (POTTS). However, these general algorithms are not efficient enough for our instances. Fortunately, the specific properties of our problem make possible the use of a considerably simpler and quicker algorithm. Several more specialized algorithms were already known (see [20] for an extended presentation) and in particular Cunningham [17] worked out a specialized algorithm to the "optimal attack problem" :"The weight of an edge represents the effort required by an attacker to destroy the edge, and the attacker derives a benefit for each new component created by destroying edges ...". (POTTS) is equivalent to the "optimal attack problem" but with a 'complementary' viewpoint : *the weight of an edge represents the benefits of cooperation between two vertices (say,*

researchers), furthermore there is a unit support for each component (say, for each research project). So there is a loss of support when two components unite. Cooperate optimally!

The algorithm described below has its roots in [17] and is similar to [18], [19]. It has the best asymptotic worst case complexity, the same as [19], and is probably the simplest, both conceptually and in the use of computational resources of the implementation. Sophisticated ingredients were necessary for finding this solution, but we don't need to make explicit use of them. The interested reader may find more details on that subject in [20].

We return now to the (POTTS) problem. An important observation is the following: if A is a set of edges, and X is the vertex set of a connected component of $G(A)$, then adding to A edges *induced* by X, that is with both endpoints in X, increases the value of f. Thus in an optimal solution A^* of (POTTS) each connected component of A^* contains all the edges of G it induces. Let us call *Potts partition* any partition of the vertices such that each class induces a connected subgraph of G.

All we have to determine is : *a Potts partition maximizing the total weight of edges induced by the classes plus the number of classes.*

As a consequence, following [19], during all the procedure it will be sufficient to consider subsets of vertices instead of subsets of edges.

A set F of edges optimizing (POTTS) will be called *optimal*, its associated Potts partition $\mathcal{P}_G(F) = \{X_1, \ldots X_k\}$ will be called *optimal* as well. For any subset F of edges let $w(F) = \sum_{e \in F} w(e)$ and for any subset X of vertices let $w(X) = \sum_{e \in E(X)} w(e)$ where $E(X)$ is the set of edges with both extremities in X.

3 Extension of a solution

In this section we will see how any set of edges maximizing (POTTS) for a graph G minus a vertex, may be completed so that it maximizes (POTTS) for the graph G itself.

Let G be a graph with a weight function w on the edges, let x be a vertex of G and let G' be the subgraph of G obtained by deleting x. The edges of G' keep their weight. Let $F' \subseteq E(G')$ be an optimal solution of (POTTS) for G'.

The following Lemma is an easy consequence of the (SUPER) property.

Lemma 1 *There exists an optimal solution of (POTTS) for G which contains all edges of F'.*

Proof: Let F be any optimal solution of (POTTS) for G. By the (SUPER) property one has in G:

$$f_G(F \cup F') \geq f_G(F) + f_G(F') - f_G(F \cap F').$$

Both F' and $F \cap F'$ are subsets of edges of G' and since F' is optimal in G', $f_G(F') = 1 + f_{G'}(F') \geq 1 + f_{G'}(F \cap F') = f_G(F \cap F')$, whence :

$$f_G(F \cup F') \geq f_G(F).$$

Since F is optimal in G by hypothesis, we get that $F \cup F'$ is optimal too, and the Lemma is proved. □

As a consequence of Lemma 1 we can obtain an optimal solution F^* for G by adding edges to F'. Let $X_1, \ldots, X_k$ be the vertices of the connected components of $G'(F')$, then $\mathcal{P}_G(F') = \{X_1, \ldots, X_k, \{x\}\}$. Each connected component in $G(F^*)$ will be either an element of $\mathcal{P}_G(F')$ or the union of at least two elements of $\mathcal{P}_G(F')$. The next Lemma shows how the value of a Potts partition is affected when a subset of connected components are put together.

Lemma 2 *Let $\mathcal{P}$ and $\mathcal{P}'$ be two Potts partitions of G such that $\mathcal{P} = (\mathcal{P}' \setminus \mathcal{W}) \cup \{\cup X_i; X_i \in \mathcal{W}\}$ for some $\mathcal{W} \subseteq \mathcal{P}$. Then*

$$f_G(\mathcal{P}) = f_G(\mathcal{P}') - (|\mathcal{W}| - 1 - w(E(\mathcal{W}))),$$

where $E(\mathcal{W})$ denotes the set of edges of G joining vertices belonging to two different sets in $\mathcal{W}$.

In particular if $\mathcal{P}'$ is optimal then $|\mathcal{W}| - 1 - w(E(\mathcal{W})) \geq 0$ and if $\mathcal{P}$ is optimal then $|\mathcal{W}| - 1 - w(E(\mathcal{W})) \leq 0$.

Proof: Replacing in $\mathcal{P}$ the sets of $\mathcal{W}$ by their union decreases the cardinality of $\mathcal{P}$ by $|\mathcal{W}| - 1$, on the other hand let A and A' be the sets of edges induced by the classes respectively of $\mathcal{P}$ and $\mathcal{P}'$, $A = A' \cup E(\mathcal{W})$ and so the weight of the edges augment by $w(E(\mathcal{W}))$. The equality is then proved and the rest follows directly. □

We are now ready to prove the following theorem:

Theorem 1 *For any* $\mathcal{W} \subseteq \mathcal{P}_G(F')$ *containing* $\{x\}$ *and minimizing* $|\mathcal{W}| - 1 - w(E(\mathcal{W}))$, *the set* $F^* = F' \cup E(\mathcal{W})$ *is optimal for (POTTS) in* G.

Proof: By Lemma 1 we know that there exists an optimal solution F^* of (POTTS) for G which contains F'. Let $\mathcal{W} \subseteq \mathcal{P}_G(F')$ such that $W = \cup_{X_i \in \mathcal{W}} X_i$ is an element of $\mathcal{P}_G(F^*)$. If $\mathcal{W}$ doesn't contain $\{x\}$ then, since F' is optimal in G', we get by Lemma 2 that $|\mathcal{W}| - 1 - w(E(\mathcal{W})) \geq 0$ (this value is the same in G and G'). On the other hand F^* is optimal in G which implies that $|\mathcal{W}| - 1 - w(E(\mathcal{W})) = 0$, but then $F^* \setminus E(\mathcal{W})$ is also optimal for G. Hence there exists an optimal solution F^* of (POTTS) for G containing F' and such that any element of $\mathcal{P}_G(F^*)$ not containing x is already in $\mathcal{P}_G(F')$.

So any $\mathcal{W} \subseteq \mathcal{P}_G(F')$ containing $\{x\}$, such that $W = \cup_{X_i \in \mathcal{W}} X_i$ induces a connected subgraph of G, and minimizing under these conditions $|\mathcal{W}| - 1 - w(E(\mathcal{W}))$, will provide an optimal solution $F' \cup E(\mathcal{W})$. (Notice that this minimum is ≤ 0 since for $\mathcal{W} = \{\{x\}\}$ we get 0.)

In order to prove the theorem it remains only to show that we may skip the connectivity constraint : we show now that any subset $\mathcal{W} \subseteq \mathcal{P}_G(F')$, $\{x\} \in \mathcal{W}$, minimizing $|\mathcal{W}| - 1 - w(E(\mathcal{W}))$ corresponds to a connected subgraph in G. Assume not, and let $\mathcal{W}_1, \ldots, \mathcal{W}_l$ $(l \geq 2)$ be the subsets of $\mathcal{P}_G(F')$ corresponding to the vertices of the connected components induced by $W = \cup_{X_i \in \mathcal{W}} X_i$ in G, $\mathcal{W}_1$ being the one containing $\{x\}$. We have

$$|\mathcal{W}| - 1 - w(E(\mathcal{W})) = |\mathcal{W}_1| + \ldots + |\mathcal{W}_l| - 1 - w(E(\mathcal{W}_1)) - \ldots - w(E(\mathcal{W}_l))$$

For $i \geq 2$, $\{x\} \notin \mathcal{W}_i$ and so as already noticed $|\mathcal{W}_i| - w(E(\mathcal{W}_i)) \geq 1$, which implies $|\mathcal{W}| - 1 - w(E(\mathcal{W})) > |\mathcal{W}_1| - 1 - w(E(\mathcal{W}_1))$ and so we get a contradiction to our assumption on $\mathcal{W}$. $\square$

At this point we see that any way to find $\mathcal{W}$ will provide a constructive algorithm for getting an optimal solution of (POTTS) : we start with a solution for a small subgraph (for example a one vertex subgraph) and then add the vertices one by one, computing at each step a new extended solution. Fortunately an optimal $\mathcal{W}$ can be computed and in the next section we are going to show how.

We remark that the value $|\mathcal{W}| - 1 - w(E(\mathcal{W}))$ doesn't depend on the subgraphs induced by the subsets X_i in $\mathcal{W}$, so we may ignore these and work in a possibly smaller graph. To be precise : the result of *shrinking* the pairwise disjoint sets $X_1, \ldots, X_k$ of $\mathcal{P}_G(F')$ in G is the graph $\text{shr}(G) = (\text{shr}(V), \text{shr}(E))$, where $\text{shr}(V) = \{x, x_1, \ldots, x_k\}$ where $x_1, \ldots, x_k$ are distinct

new vertices, and the function shr $: V \longrightarrow \{x, x_1, \ldots, x_k\}$ is defined with $\mathrm{shr}(v) := x_i$ if $v \in X_i$ and $\mathrm{shr}(x) := x$; the image of an edge e with extremities x and y is $\mathrm{shr}(e)$ with extremities $\mathrm{shr}(x)\mathrm{shr}(y)$, moreover, if this is a loop $(\mathrm{shr}(x) = \mathrm{shr}(y))$, it is deleted; edges keep their weight, that is $w_{\mathrm{shr}}(\mathrm{shr}(e)) = w(e)$; sets of vertices or of edges are replaced by the image sets. There is a one to one correspondence between the subsets $\mathcal{W}$ of $\mathcal{P}_G(F')$ containing $\{x\}$ and subsets W of vertices of $\mathrm{shr}(G)$ containing x and for any such $W = \mathrm{shr}(\mathcal{W})$ one has $|\mathcal{W}| - 1 - w(E(\mathcal{W})) = |W| - 1 - w_{\mathrm{shr}}(W)$ (as defined in the preceding chapter $w_{\mathrm{shr}}(W)$ is the sum of weights of the edges in the subgraph of $\mathrm{shr}(G)$ induced by W). So $\mathcal{W}$ will be optimal if and only $W = \mathrm{shr}(\mathcal{W})$ minimizes $|W| - 1 - w_{\mathrm{shr}}(W)$.

4   A network flow model

Given a graph H with weight function w on the edges and $W \subseteq V(H)$, let $b(W) = |W| - 1 - w(W)$. From the previous chapter it is clear that any way of solving the following problem will provide an algorithm solving (POTTS) : given a vertex x in H, find W^* a subset of vertices of H containing x such that $b(W^*) = min(b(W); W \subset V(H), x \in W)$. This problem is solved both in Picard-Queyranne [24] and Padberg-Wolsey [25], using a network flow model that will be described below. (The work [24], [25] solve much more complex problems involving several network flow computations – the subroutine below is the adaptation of an auxiliary procedure.)

We first give some definitions and wellknown facts. Given a directed graph and a subset X of its vertices, $\delta(X)$ will denote the set of arcs leaving X, and is called a *cut*; if $s \in X$, $t \notin X$ it is an (s, t)-cut; $\delta(X, Y)$ denotes the set of arcs oriented from X to Y. A function c with nonnegative value on the arcs is called a *capacity* function. If F is a set of arcs then $c(F) := \sum_{e \in F} c(e)$. A *network* is a directed graph with capacity function. By a well known theorem of Ford and Fulkerson [26] a minimum (s, t)-cut in a network may be found by computing a maximum flow in this network. Very efficient maximum flow algorithms are known [27].

We now describe in several steps a network $(D, c) = N(H, w)$ that will be associated to H and w:
1. $V(D) := V(H) \cup \{s, t\}$ where s, t are distinct new vertices;

2. Define for all $u \in V(H)$:

$$p(u) := 1/2 \sum_{v \in V(H),\, uv \in E(H)} w(uv), \text{ and then}$$

  – if $p(u) > 1$ add an arc (s, u) of capacity $c(s, u) = p(u) - 1$,

  – if $p(u) < 1$ then add an arc (u, t) of capacity $c(u, t) = 1 - p(u)$.

3. to each edge uv of $E(H)$ we associate the arcs (u, v) and (v, u) of capacities $c(u, v) = c(v, u) = \frac{1}{2}w(uv)$.

Let $W \subseteq V(H)$, $\delta(\{s\} \cup W)$ is an (s, t)-cut of $N(H, w)$ and reciprocally any (s, t)-cut of $N(H, w)$ corresponds to a subset of vertices of H. There is a close relationship between the capacities of the (s, t)-cuts of $N(H)$ and the values of b on subsets of vertices of H :

$$c(\delta(\{s\} \cup W)) = |W| - w(W) + K = b(W) + K + 1, \qquad (CUT)$$

where $K := c(\delta(s)) = \sum_{(s,x) \in A} c(s, x)$.

Indeed, let us see how the capacity of the cuts changes if we start with the set $\{s\}$ inducing an (s, t)-cut of capacity $c(\delta(s)) = K$ and then 'add' to it the vertices of W one by one:

The contribution of adding v to the side of s is $1 - p(v)$, for either it decreases from $p(v) - 1$ to 0 (this happens if $p(v) > 1$ see the first case of '2.' in the construction of $N(H, w)$), or it increases from 0 to $1 - p(v)$. Thus the contribution of these arcs is:

$$\sum_{v \in W} 1 - p(v) = |W| - w(W) - 1/2 \sum_{xy \in E(H),\, x \in W,\, y \notin W} w(xy).$$

On the other hand the contribution of the arcs between W and $V(H) \setminus W$ is clear: at the beginning it is zero, and at the end it is:

$$c(W, V(H) \setminus W) = 1/2 \sum_{xy \in E(H),\, x \in W,\, y \notin W} w(xy).$$

The change comparing to K is provided by the sum of the two contributions, which is $|W| - w(W)$, and so equation (CUT) has been proved.

11

From (CUT) we get that any (s,t)-cut $C = \{s\} \cup W$ containing the special vertex x and of minimum capacity will correspond to our goal : a subset W of vertices of H containing x and minimizing b. Such a cut is easy to obtain : shrink $\{s,x\}$ or equivalently, add to $N(H,w)$ an arc of infinite capacity from s to x, and compute in the new network a minimum (s,t)-cut : this cut will contain x and will be minimum among (s,t)-cuts containing x.

Let us note that this method can be easily *generalized to minimize any modular shift of* $-w(W)$. Adding $|W| - 1$ is just a particular choice of a modular function. That is, one can minimize $(\sum_{w \in W} m(w)) - w(W)$, where $m : V \longrightarrow \mathbb{R}$. Indeed, for generalizing from $|W|$ to $m(W)$ one only has to write $m(x)$ instead of 1 in both cases of Step 2 in the construction of N(H,w).

5   The Algorithm

In this section we state the 'Optimal Cooperation Algorithm' whose validity is a consequence of the results of the two preceding chapters.

At each iteration a subset $U \subseteq V$ and a partition $\mathcal{P}$ of U will be at hand. In Step 0 we give trivial initial values; in Step 1, we choose an arbitrary vertex u to be added to U and through the following steps we *compute a subset* $\mathcal{W}$ *of* $\mathcal{P} \cup \{u\}$ *providing a partition of* $U \cup \{u\}$ *consisting of the sets contained in* $\mathcal{P}$ *but not in* $\mathcal{W}$ *plus a new class containing all the other vertices (that is : the new class is the union of all sets in* $\mathcal{W}$*). The set of edges corresponding to this new partition is the union of the set of edges corresponding to the previous one and the edges between the sets in* $\mathcal{W}$*.

OPTIMAL COOPERATION ALGORITHM

INPUT: A graph $G = (V, E)$, and a weight function $w \colon E \longrightarrow (0,1)$, where $(0,1) := \{t \in \mathbb{R} : 0 < t < 1\}$.

OUTPUT: A partition $\mathcal{P}^*$ optimizing (POTTS) : the set of edges A^* induced by the sets in $\mathcal{P}^*$ maximizes $\{f_{G,w}(A) := c_G(A) + \sum_{e \in A} w(e) : A \subseteq E\}$.

0. $U := \emptyset$, $\mathcal{P} := \emptyset$.

Do n times consecutively Steps 1 to 5, and then define $\mathcal{P}^* = \mathcal{P}$:

1. Choose a vertex $u \in V \setminus U$.

2. Define H and w_H as the result of shrinking the classes of $\mathcal{P}$ in $G(U \cup \{u\})$ with weight function w restricted to the edges of $G(U \cup \{u\})$.

3. Construct $N(H, w_H)$, add an arc of "infinite" capacity from s to u and using any maximum flow algorithm compute, in the so obtained network, a minimum (s, t)-cut defined by the set of vertices $C = \{s, u, x_1, \ldots, x_k\}$ where each x_i is a vertex of H corresponding to a set $X_i \in \mathcal{P}$.

4. Define $R := \{u\} \cup X_1 \cup \ldots \cup X_k$

5. Redefine U and $\mathcal{P}$: $U := U \cup \{u\}$, $\mathcal{P} := (\mathcal{P} \setminus \{X_1, \ldots, X_k\}) \cup \{R\}$;

END;

Remarks:

- "Infinite" means big enough to avoid a minimum cut containing this arc: for example the sum of the capacities of all arcs.

- Note that C can be equal to $\{s, u\}$.

- The graphs H can also be constructed iteratively with only one shrinking in each iteration, by adding u and then shrinking the set $C \setminus \{s\}$.

- The choice for u is completely free; this freedom could be used for making the computations easy.

Theorem 2 *The output $\mathcal{P}^*$ is an optimal partition for (G, w).*

Proof: At step 0, $\mathcal{P} := \emptyset$ is an optimal solution for the subgraph of G induced by $U := \emptyset$. Assume now that $\mathcal{P}$ is an optimal solution for the subgraph of G induced by U, and show that after applying steps 1 to 5 the new partition is optimal for $U \cup \{u\}$. From the preceding chapter we know that $W := C \setminus \{t\} = \{u, x_1, \ldots, x_k\}$ is a subset of vertices of H containing u minimizing $|W| - w_H(W) - 1$. But H and w_H are obtained by shrinking the classes of $\mathcal{P}$ in $(G(U \cup \{u\}), w)$, hence $\mathcal{W} = \{u, X_1, \ldots, X_k\}$ is a subset of $\{u\} \cup \mathcal{P}$ minimizing $|\mathcal{W}| - 1 - w(E(\mathcal{W}))$ and by Theorem 1 the partition $(\mathcal{P} \setminus \{X_1, \ldots, X_k\}) \cup \{R\}$ is then optimal $\quad \square$

Notice that the "Optimal Cooperation algorithm" consists merely of $n-1$ network flow computations and $n - 1$ shrinking.

6   Implementation and evaluation

In this section we discuss some aspects of the practical implementation of the optimal cooperation algorithm. Here we consider a two dimensional square

lattice where the weights, w, follow a bimodal distribution:

$$P(w) = p\delta(w - w_1) + (1 - p)\delta(w - w_2) \tag{7}$$

where $\delta(x) = 1$ if $x = 0$ and $\delta(x) = 0$ otherwise. We restrict ourself to the symmetric case $p = 1/2$. As explained in [11] in the infinite lattice limit there is a second-order phase transition in the model at $w_1 + w_2 = 1$. In the strongly disordered phase, $w_1 + w_2 \ll 1$ the optimal set of edge is $A = \emptyset$, while in the strongly ordered phase, $w_1 + w_2 \gg 1$ it is $A = E$. In between, the optimal set A has a non-trivial structure and at the transition point it is a fractal reflecting the critical properties of the system[11].

As explained in the previous section, for a graph with n vertices the algorithm consists in $n - 1$ steps. At each step a new site is incorporated to the current graph for which the optimization has already been performed. There is a freedom of the order of the added sites, which can be used to speed up the computation, as well as to obtain the optimal sets of intermediary lattices. For example from the computation of the optimal set for an $L \times L$ square lattice with free boundary conditions one can also obtain the optimal sets of $L - 1$ minors of size ranging from 1 to $L - 1$ with free boundary conditions, which is useful for finite size analysis.

The "engine" of the algorithm is the max flow solver, which is called at each step. We have used the Goldberg and Tarjan algorithm [28]. Another important point concerns the building of the network (point 3 in the Optimal Cooperation Algorithm). It is possible to build it explicitly as another data structure than the lattice. Alternatively one can increase up to a sufficiently large value all the capacities of the edges of a spanning tree of each connected component as mentioned in the previous paragraph. In general the cpu time required to explicitly shrink the connected components is larger than the cpu time needed to simply increase some capacities. On the other hand, keeping all vertices will increase the size of the network in which the max flow has to be found. If all the capacities are large ($w_1 + w_2 > 1$, that is we are in the ordered phase) the network will have only two sites at each step, and therefore the effort to build the network will be largely compensated by the speed at which the maximum flow is found. In the opposite extreme, when the optimal set tends to be empty, it is questionable which of the two strategies is the best. We have found that explicitly shrinking the connected components is always the most efficient. As a matter of fact the computation time is about

L	time
16	$\sim$ one second
32	$\sim$ half a minute
64	$\sim$ five minutes
128	$\sim$one hour and a half
256	$\sim$ one day
512	$\sim$ two weeks

Table 1: Typical cpu time (see text)

ten times larger at the critical point $w_1 + w_2 = 1$ than at the ordered phase $(w_1 + w_2 > 1)$ and about three times larger than at the disordered phase $(w_1 + w_2 < 1)$.

As an illustration we display in the table below some typical cpu times needed to find an optimal set on a periodic square lattice of various size, L, at the critical point $w_1 + w_2 = 1$. From these results one can extract the empirical rule that in our algorithm the computation time grows approximately as the fourth power of the linear size L, thus as the second power of the number of sites. These cpu times refer to a PC with a pentium III, 800 megahertz processor.

In conclusion in this paper we have presented and implemented a combinatorial optimization algorithm which solves the optimal cooperation problem in polynomial time. The procedure, which can be used to obtain the partition function of the q-state random bond Potts model in the large q-limit is, up to our knowledge, the first application of submodular function optimization in (statistical) physics. Detailed results about the q-state random bond Potts model, such as properties of phase transitions in two- and three-dimensional lattices will be presented in a separate publication[29].

References

[1] J.M. Yeomans, *Statistical Mechanics of Phase Transitions*, Clarendon Press, Oxford (1992).

[2] R.J. Baxter, *Exactly solved models in Statistical Mechanics*, Academic Press, New York (1982).

[3] F. Barahona, R. Maynard, R. Rammal, J. P. Uhry, *Morphology of ground states of a two-dimensional frustration model*, J. Phys. **A 15**, 673–679 (1982).

[4] I. Bieche, R. Maynard, R. Rammal and J.P. Uhry, *On the ground states of the frustration model of a spin glass by a matching method of graph theory*, J. Phys. **A 13**, 2553-2576, (1980).

[5] J. C. Anglès d'Auriac, R. Maynard, *On the random antiphase state in the ±spin glass model in two dimensions*, Solid State Communications **49**, 785 (1984).

[6] A. Galluccio, M. Loebl, J. Vondrák, *New Algorithm for the Ising Problem: Partition Function for Finite Lattice*, Graphs, Physical Rev. Letters, **84**, 5924–5927 (2000).

[7] J. C. Anglès d'Auriac, M. Preissmann, R. Rammal, *The random field Ising model : algorithmic complexity and phase transition*, J. Physique Lett. **46,** 173 (1985).

[8] J. C. Anglès d'Auriac and N. Sourlas, *The 3-d Random Field Ising Model at zero temperature*, Europhys. Lett. **39**, 473 (1997).

[9] J.C. Anglès d'Auriac, M. Preissmann and A. Sebő, *Optimal cuts in graphs and statistical mechanics*, Journal of Math. and Computer Modelling **26,** 1 (1997).

[10] M.J. Alava, P.M. Duxbury, C.F. Moukarzel and H. Rieger, Exact combinatorial algorithms: Ground states of disordered systems, in *Phase Transitions and Critical Phenomena*, Vol. 18, p 143 (C. Domb and J.L. Lebowitz, eds), (London, Academic, 2001).

[11] R. Juhász, H. Rieger and F. Iglói, *Random bond Potts model in the large-q limit*, Phys. Rev. E **64**, 56122 (2001).

[12] M. Grötschel, L. Lovász, A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer Verlag, Berlin Heidelberg (1988).

[13] A. Schrijver, *A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time*, Journal of Combinatorial Theory, Series B, **80**, No. 2, 346-355 (2000).

[14] S. Iwata, L. Fleischer, and S. Fujishige, *A combinatorial strongly polynomial algorithm for minimizing submodular functions*, Journal of the ACM **48**, Issue 4, 761-777 (2001).

[15] S. Iwata, *A Fully Combinatorial Algorithm for Submodular Function Minimization*, Journal of Combinatorial Theory, Series B **84**, No. 2, 203-212 (2002).

[16] A. Frank, É. Tardos, *Generalized Polymatroids and Submodular Flows*, Mathematical Programming 42, 489-563 (1988).

[17] W. H. Cunningham, *Optimal Attack and Reinforcement of a Network*, Journal of the ACM **32**, No 3, 549-561 (1985).

[18] F. Barahona, *Separating from the dominant of the spanning tree polytope*, Operations Research Letters **12**, 201-203 (1992).

[19] M. Baïou, F. Barahona, A. R. Mahjoub, *Separation of Partition Inequalities*, Math. of OR, **25**, 243–254 (2000).

[20] F. Iglói, M. Preissmann, A. Sebő, *Optimal Cooperation*, Cahiers du laboratoire Leibniz (2002).

[21] F.Y. Wu, *The Potts model*, Rev. Mod.Phys **54**, 235 (1982).

[22] P.W. Kasteleyn and C.M. Fortuin, J. Phys. Soc. Jpn **26** (Suppl.), 11 (1969).

[23] L. Lovász, *Combinatorial Problems and Exercises*, North Holland and Akadmiai Kiadó, (1979).

[24] J.-C. Picard, M. Queyranne, *Selected Applications of Minimum Cuts in Networks*, INFOR 20, 394-422 (1982).

[25] M. W. Padberg, L. A. Wolsey, *Trees and Cuts*, Annals of Discrete Mathematics **17** 511-517 (1983).

[26] L.R. Ford and D.R. Fulkerson, *Maximum flow through a network*, Canad. J. Math **8** 399–404 (1956).

[27] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, Network flows: Theory, Algorithms and Applications, Prenctice-Hall, 1993.

[28] A.V Goldberg and R.E. Tarjan, *A new approach to the maximum-flow problem*, Journal of the Association for Computing Machinery **35**, 921–940, (1988).

[29] J.C. Anglès d'Auriac, F. Iglói, M. Preissmann, A. Sebő, (unpublished).